DOCUMENT RESUME

ED 037 697 AL 000 812

AUTHOR	Moyne, J. A.
TITLE	Towards the Understanding of Natural Languages by Machines.
INSTITUTION	International Business Machines Corp., Cambridge, Mass. Boston Programming Center.
REPORT NO	TR-BPC-2
PUB DATE	2 Oct 67
NOTE	9p.; Paper read at the International Congress of Linguists, Bucharest, Romania, August 28-September 2, 1967
EDRS PRICE	EDRS Price MF-$0.25 HC-$0.55
DESCRIPTORS	*Computational Linguistics, Computer Programs, Data Bases, Deep Structure, Dictionaries, Phonology, *Programing Languages, Semantics, Surface Structure, *Transformation Generative Grammar
IDENTIFIERS	Proto RELADES, *Recognition Grammar

ABSTRACT

At present a computer system cannot be constructed for handling the totality of a natural language in any significant way. It is, however, possible to construct a system for communication in a narrow field of discourse. A working model for a specialized discourse based on a recognition grammar is discussed. Some properties of the recognition grammar, which is based on transformational theory, are outlined. (Author/FWB)

TECHNICAL REPORT RPC 2

October 2, 1967

TOWARDS THE UNDERSTANDING OF NATURAL LANGUAGES BY MACHINES

J. A. Moyne

ABSTRACT

At present a computer system cannot be constructed for handling the totality of a natural language in any significant way. It is, however, possible to construct a system for communication in a narrow field of discourse. A working model for a specialized discourse based on a recognition grammar is discussed. Some properties of the recognition grammar, which is based on transformational theory, are outlined.

This paper was read at the International Congress of Linguists at Bucharest, August 28 - September 2, 1967. It is being published in the proceedings of the Congress. This paper is, therefore, not available for public distribution.

IBM

Boston Programming Center

545 Technology Square

Cambridge, Mass. 02139 (U.S.A.)

TOWARDS THE UNDERSTANDING OF NATURAL LANGUAGES BY MACHINES*

J. A. Moyne

International Business Machines Corporation

1. Recognition Grammars

A generative transformational grammar has three parts: (1) A base component that generates deep structures. The deep structure is an abstract and complex object that carries the meaning of the sentence. (2) Transformational rules that apply to the deep structure and produce a surface structure similar to a traditional parsing. (3) Phonological rules that apply to the surface structure and produce a phonological representation.

For a computer to "understand" a natural-language sentence, the generative grammar must be reversed: a surface phrase-structure grammar parses the input sentence producing a surface structure; reverse transformations apply to the surface structure and produce a deep structure (cf. 4.); and, finally, semantic rules "interpret" the deep structure as actions to be performed by the machine. Such a reverse grammar is called a recognition grammar.

---

*For discussions of the theoretical foundations upon which this work is based see Noam Chomsky, Aspects of the Theory of Syntax, The MIT Press, Cambridge, 1965; J. A. Fodor and J. J. Katz (eds.), The Structure of Language: Readings in the Philosophy of Language, Englewood Cliffs, N. J., Prentice-Hall 1964; Jerrold J. Katz and Paul M. Postal, An Integrated Theory of Linguistic Descriptions, The MIT Press, Cambridge, 1964; and the references contained in the above works. I am grateful, among others, to G. Carden, R. Carter and N. Rochester for discussions and many editorial improvements on this paper. D. B. Loveman designed the computer programming language which is used in developing the system described in this paper.

There has been a great deal of work on generative transformational grammars, but very little on recognition grammars. In this paper I shall discuss some hypotheses about recognition grammars and describe a working computer system that "understands" English within a limited universe of discourse.

2. Proto-RELADES

Since a recognition grammar faces the same unsolved problems as a generative grammar, I do not think that we can, as yet, build a computer system to process and understand the whole of a natural language. We can, however, build a practical system to handle inquiries about a given subject or data base.

Proto-RELADES is such a system. It uses an IBM System/360 computer to communicate with a library in English, but its primary purpose is to experiment with communication with computers in natural languages. Since we hope to expand the system to handle other data bases and more sentence patterns, we have attempted to make both the control system and the grammar as general as possible, avoiding ad hoc solutions even at the cost of inefficiency within the library data base. We believe the present system can be expanded within the limitations of current linguistic theories.

Proto-RELADES has a small dictionary, a transformational recognition grammar, and a semantic interpreter controlling computer operations. If the input sentence is "Give me the list of any books you have about grammar," the system will analyze and "understand" the sentence, and supply the list requested.

The programs that operate the dictionary and grammar are independent of any particular grammar, data base, or language. Thus a recognition grammar of any language could be plugged into the system with little or no change in programming. Alternately, the existing English grammar could analyze sentences

about other subjects-"What can you tell me about the current Middle-East dispute?" or "Solve the following equations:..." To get the desired response to such sentences, we must supply the dictionary with any missing words and give the computer programs and data to carry out the necessary commands. Proto-RELADES can also be reversed to test transformational or phonological rules, accepting any deep structure and set of rules as input and producing a surface structure.

3. The Proto-RELADES Grammar

The recognition grammar in Proto-RELADES is a reversed transformational grammar with four components: lexicon, surface grammar, deep grammar, and semantics.

The lexicon contains the vocabulary of the discourse about the data base, in this case library operations. It also determines syntactic category from context and replaces idioms with single-word synonyms: "having to do with" = "concerning."

The output of the lexicon is the bottom two lines of the surface structure tree; example:

Input Sentence: Have any books about grammars been written?

TNS	HAVE	ART	ADJ_1	ART	NU	N	P	ART	NU	N	TNS	BE	TNS	VT_3	Q
PRST	have	∅	any	∅	PL	book	about	∅	PL	grammar	EN	be	PAST	write	?

Figure 1

The surface grammar is an inverse phrase structure grammar with rules of the form R: $A \leftarrow Y$ (R is a rule label, A a single element, Y a string):

Rule 10: NP $\leftarrow$ DET NU N

Context-sensitivity is achieved indirectly by letting rules call other rules;

the CS rule A⟵Y // X __ Z would be written in two steps as:

Rule Ni: Rule Sj ⟵ XYZ

Rule Sj: A⟵Y

If Y is in the context X __ Z when rule Ni applies, rule Sj is called and rewrites Y as A. Rule S(j+1) can then return control to rule N(i+1). If the context is not satisfied, rule Sj will never apply.

The surface grammar is divided into partitions in which rules apply cyclicly to their own output. When no more rules can apply in one partition, control passes to the next. At the end of the last partition, control returns to the first rule of the first partition. This double-cycle ordering saves computer time and prevents unnecessary blocked analyses.

The output of the surface grammar is the surface structure tree; example:

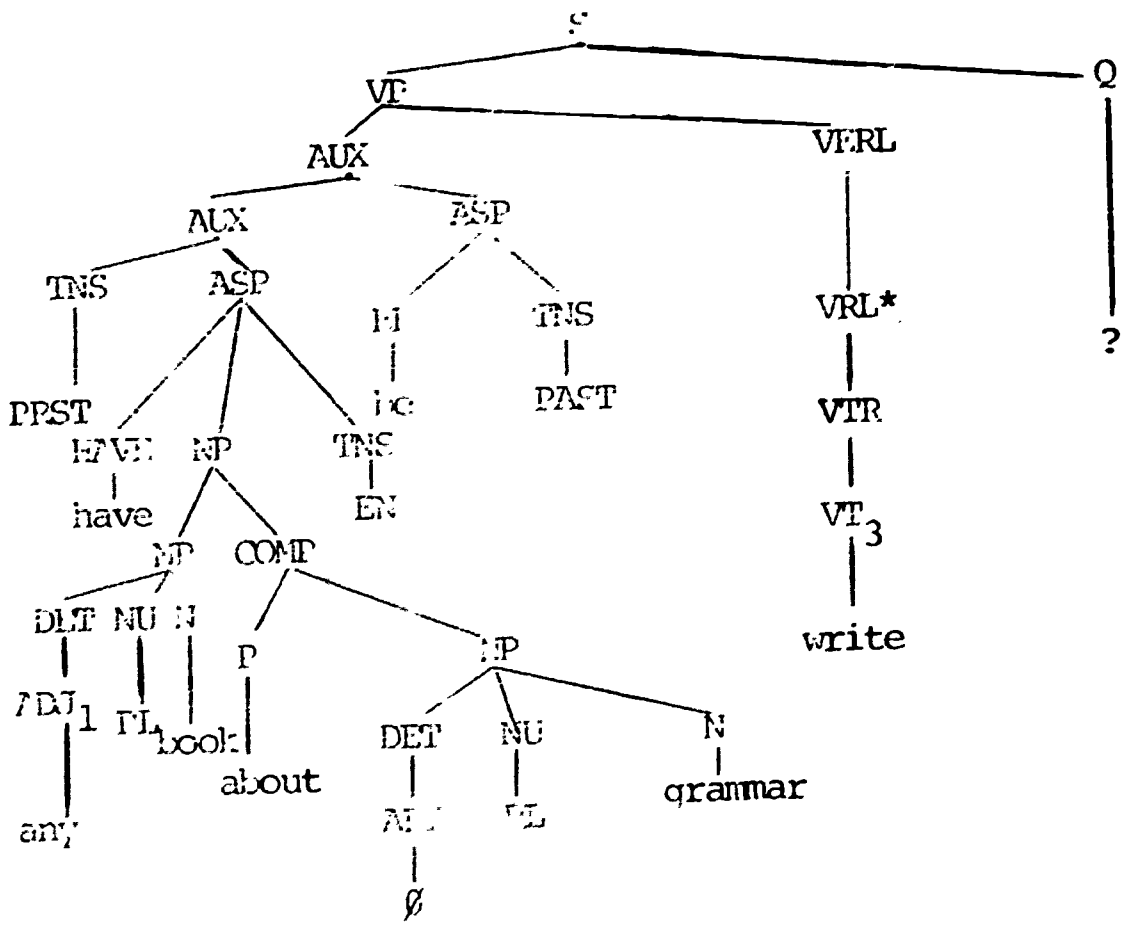

Figure 2

(this and other figures in this paper have been slightly modified from actual computer outputs for the sake of simplicity.)

The deep grammar is a set of ordered transformational rules. Each rule has two parts: a structural description defining the surface or intermediate structure subject to the rule, and computer instructions that make the desired changes. If a rule does not apply, the next rule in the sequence is tried; if a rule applies, control may be transferred to the next rule, back to the rule just applied (for an iterative rule), back to an earlier rule (to re-apply a sequence of rules), or on to a later rule (to skip over rules that will not apply). This freedom in ordering saves computer time. When all the applicable transformations have applied, the resulting tree is the deep structure of the input sentence:

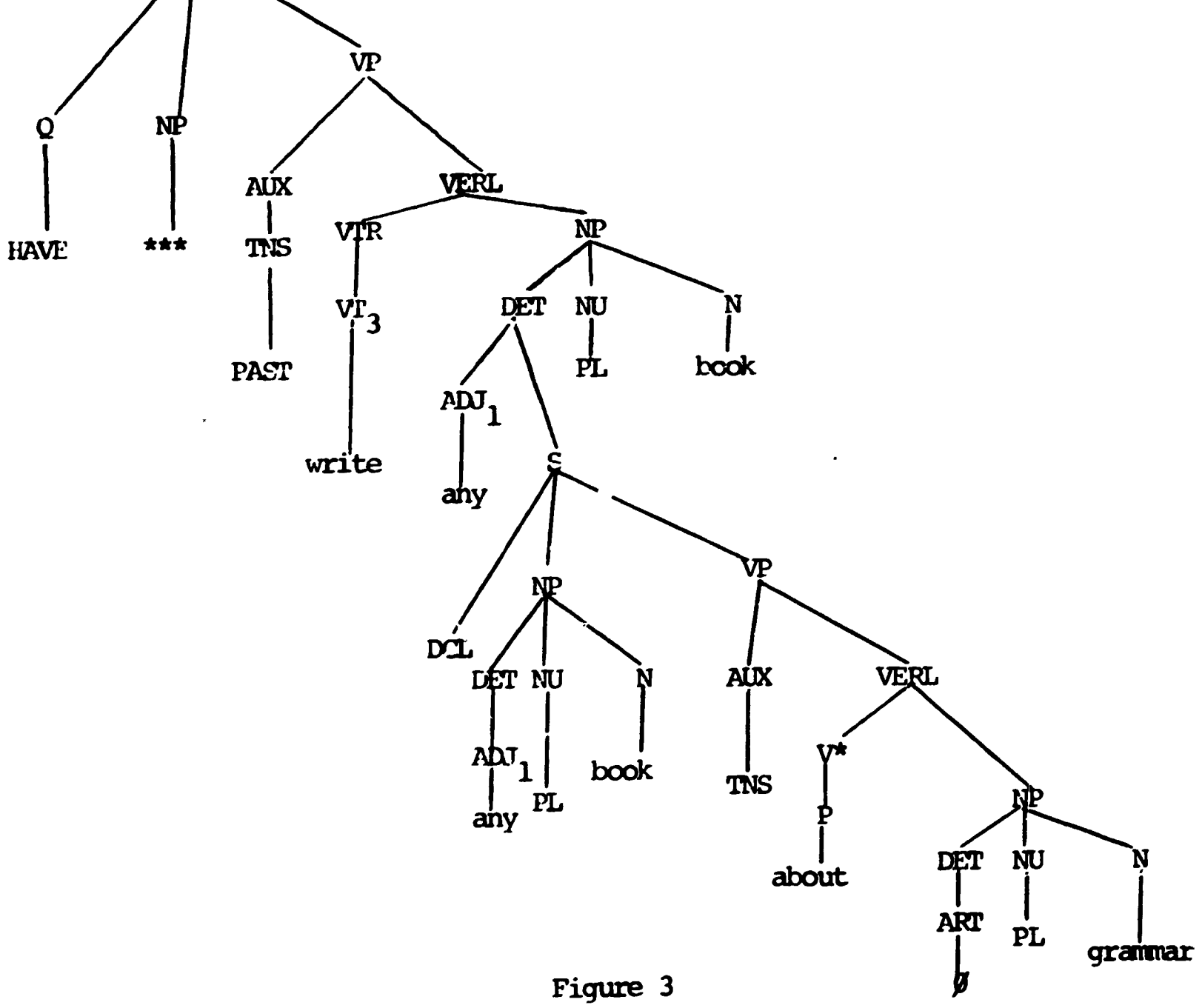

Figure 3

(*** in this figure represents a dummy subject generated by the grammar)

In Proto-RELADES all deep structures are based on this form:

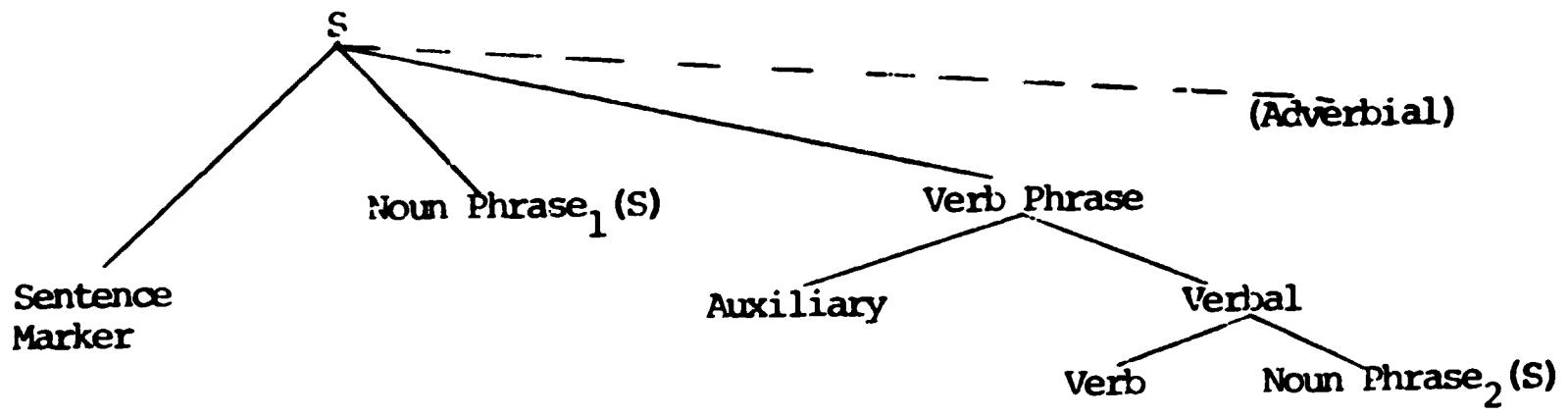

Figure 4

Another sentence (S) with the same structure may be embedded under each NP, and so on indefinitely. Unlimited recursion cannot lead to problems in a recognition grammar because the depth of embedding is controlled by the input sentence.

It is interesting that this simple deep structure is adequate for semantic interpretation without appeal to hypothetical verbs or implausible embeddings. Such a level, intermediate between surface structure and the highly abstract output of a generative semantics or other base components, seems to be necessary for computerized recognition grammars, and might well be useful in a model of perception or learning.

4. Ambiguity Problems

Computers are notorious for finding ambiguities in the simplest sentences; Proto-RELADES solves this problem by its restricted data base and discourse. Lexical ambiguities are resolved by giving words only the meaning pertinent to the data base, and some structural ambiguities can be handled similarly: "List books on computers in the library", for our library system, can only mean books in the library about computers, not books on top of computers located in the library. Data-base restrictions therefore permit the first analysis and reject

the second.

Some sentences remain ambiguous even within the restricted data base; we propose to resolve these ambiguities by conversation between man and machine. If the input is: "List the documents about books in the library." the computer produces the possible relevant analyses and asks the user, "Do you mean documents about the library's books or documents in the library about books." The user answers and the machine executes the approved analysis. This device is not yet working.

This system, with a restricted data base and man-machine conversation, is less ambitious than programs that permit all possible analyses and try to select the relevant one. Our model may be closer to human analysis, which is also controlled by context and resolves ambiguities by questioning.

5. Semantics

The semantic component of Proto-RELADES consists of ordered transformational rules that apply to a deep structure and produce an executable statement:

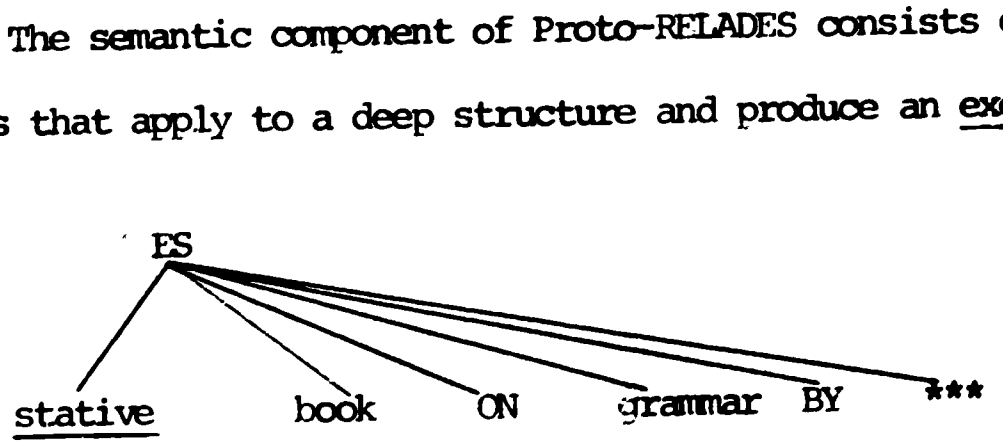

Figure 5

The system then calls an operational program to execute that statement. The semantic transformations depend on the data base, but the control program that applies them is completely independent.

We originally intended to write a separate operational program to represent the meaning of each of the dozens of verbs in the Proto-RELADES lexicon, and

have a control program to select which operational programs applied and to arrange them in the proper order. We think this is more practical than the usual method of translating the analysis of the English input into an artificial language.

To our surprise we found that we needed only two programs as semantic primitives for the whole library lexicon. The stative program prints "N documents were found," the non-stative program prints a list. Verbs that ask whether the library has a certain document are [+stative]. All other verbs in this system turn out to be requests for information from the library catalog and are [-stative]. Normally [-stative] verbs like "write" ("Write the list of...") will call the stative program in appropriate context ("Are there any books written about computers?").

Obviously a more complex data base would require more operational programs, but we are convinced that a powerful system with many applications could be built with a reasonable number of operating programs.

Our experience with Proto-RELADES thus suggests that semantic primitives may not be atomistic markers but rather complex and powerful entities like our stative and non-stative programs. Perception would then work by equating words with these complex primitives through transformational rules in the semantic component. We feel that this speculation merits further investigation.